Library Database Application

---A CMPT 354 Mini Project

DIVYA THOMAS

QIANHUI XU

April 7, 2020

Project Specifications

We have assumed that the database is for a university library but it can be used for any library. Our library database has the following characteristics:

- Library has Items. All **Items** in the library are of three kinds- books, periodicals (journals, magazines etc.), audiovisuals (CD, DVD, cassette, records etc). Items are identified by an item ID. Other information such as title of the item, date of when it was produced, shelf where it is located in the library (shelf_num), whether it is currently available or not, if loaned- then the last borrow date, due date and if returned- then last return date are also stored. Each item also belongs to one of 7 categories namely Books-nonfiction, Books-fiction, Journal, Magazine, Movies, Music, Others.
- A **book** can also have an edition and can indicate if it is available online or not. It can be authored by many authors. Each author is identified by an author ID and has an author first and last name.
- **Periodicals** can have an issue number, volume and if it is available online or not. It can also have many publishers. Each publisher is identified by an ID and has a publisher name.
- **Audiovisuals** have a type of medium (DVD, cassette, records, other) and can have multiple artists and producers. Each artist is identified by an ID and has an artist name. Each producer is identified by an ID and has a producer name
- A **Person** is identified by an ID and has a first name, last name, phone, email, address, gender (M/F/O), member type (since we are assuming a university library, this will be staff, faculty, undergraduate, graduate student), membership start date, membership end date and a Boolean value indicating whether the person is currently allowed to borrow items. A **library personnel** is a type of person who additionally has a job start date and end date
- A person is automatically subject to fines if they do not return items by the due date. **Fines** are identified by a combination of item ID, person ID and the fine date. Fines also have the amount of fine due and the amount paid so far. A person can have many fines on the same item (borrowed and returned late many times) and many people can have fines on the same item. We have set our due date to be 2 months from the date borrowed and fines will be $2 per late day.
- A person can also volunteer for the different volunteering activities offered by the library. Each **volunteering opportunity** is identified by a unique name and also belongs to one of 5 types of volunteering activity categories (Adult literacy tutoring, Adult numeracy tutoring, Home delivery of library materials, Teen volunteers, others). Each activity also has a short description and an estimated hours per week needed.
- A person can also request help from the library. Each **help request** is identified by an ID. A person can choose one of the 5 different types of help request categories to place their help request into, namely Research/finding items, Citation help, Writing, Event/workshop help, Others. Each help request is also accompanied by a brief description written by the

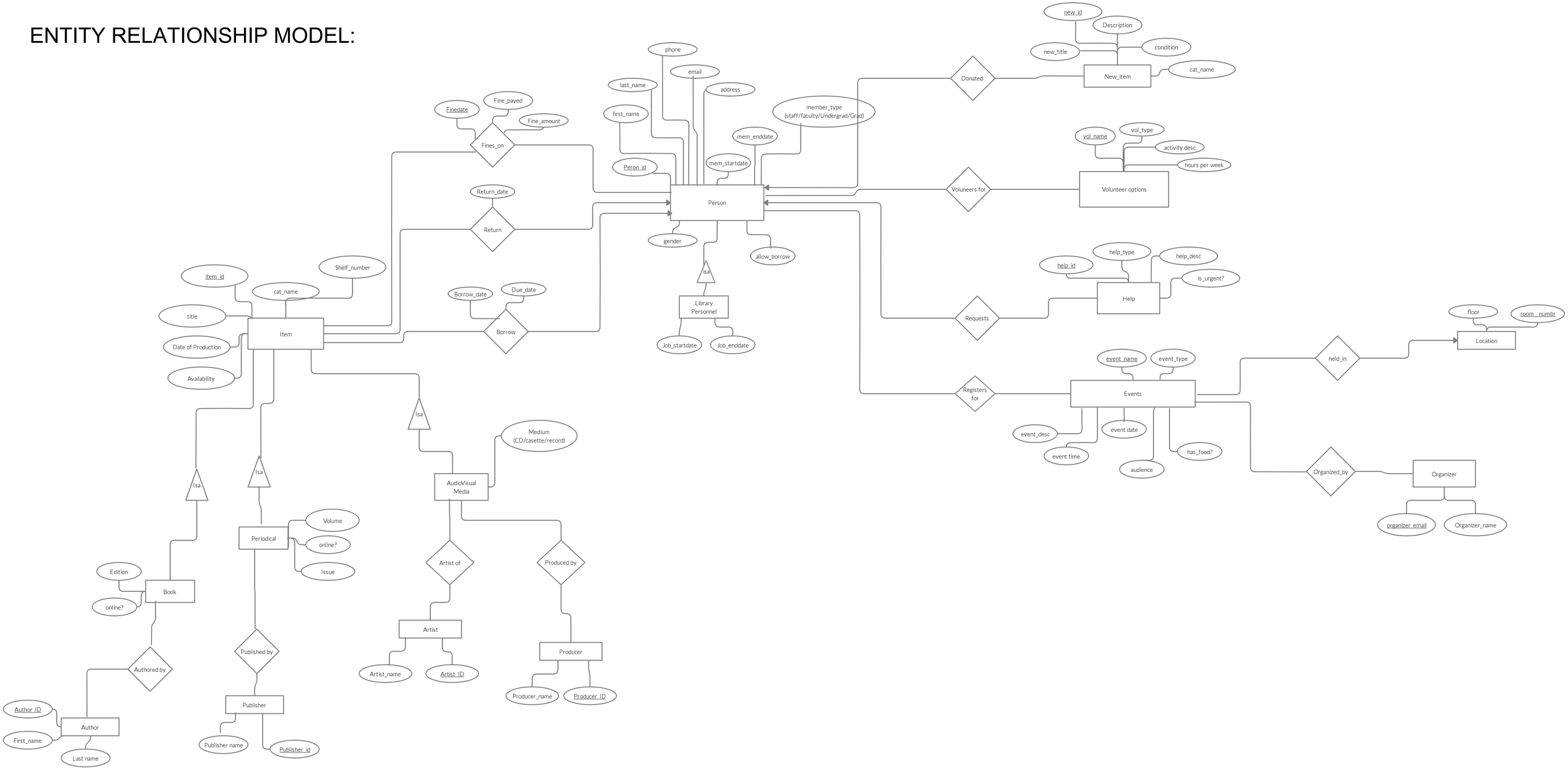
user which gives the library more details about the request. The person can also choose to indicate whether the request is urgent or not.

- A person can also donate items to the library. These might be added to the library in future after librarians review them (i.e they are not automatically added to library items). This is because maybe the item may not be in good condition or maybe too redundant. These **new items** are identified by an ID which is automatically assigned by the system. Each person can provide a title, short description and condition of the item. The person can also choose one of 7 categories namely Books-nonfiction, Books-fiction, Journal, Magazine, Movies, Music, Others to which the item most belongs. A person can donate many items but each item is donated by a specific person.
- A person can register for many events happening in the library. Each **event** is identified by a unique event name, event type (book related events, art shows, film screenings, workshops, networking, others), event description, event time, event date, recommended audience, a Boolean value indicating if the event has food or not, location and organizers. There is no fee to attend any event.
- Each event is organized by a single organizer or multiple organizers. Each **organizer** is identified by a unique organizer email and has an organizer name. Each organizer can host multiple events.
- Each library **location** (social rooms) is identified by a unique room number and the floor number of this room is also specified. Each event is held in a specific room.
- A library user can borrow many items and return many items. But each item can only be borrowed and returned by one person at a time.

An ERM has been designed based on these characteristics and can be found on the next page.

Our database application which uses this database allows a library user to find, borrow, return, donate items, find and register for events and also seek help from the library.

ENTITY RELATIONSHIP MODEL:



Converting ERM to Relations and checking for Anomalies

The creation of the following relations are proposed based on ERM:

Entity Sets and one-many relations

The entity sets in ERM were converted to tables. In one-many relationships, the primary key of the 'one' entity was added as foreign key to the 'many' entity and also any attributes of the one-many relationship. The proposed relations are as follows:

- Item(item_id, cat_name, title, pub_date,availability, shelf_num, Person_ID[FK-Person] ,borrow_date, due_date,return_date)
- Person(Person_ID, first_name, last_name, address1, address2, city, state, zipCode, phone, email,gender,member_type, mem_startdate, mem_enddate,allow_borrow)
- Personnel(Person_ID[FK-Person], job_startdate, job_enddate)
- Books(item_id[FK-Item], edition, online)
- Periodicals(item_id[FK-Item] , issue, volume,online)
- Audiovisuals(item_id[FK-Item] , medium)
- Artist(artist_id, Artist_name)
- Author(author_id, first_name, last_name)
- Publisher(publisher_id, Publisher_name)
- Producer(producer_id, Producer_name)
- Volunteer_options(vol_name, vol_type, activity_desc ,hours_week)
- Help(help_id, help_type, help_desc, is_urgent, Person_ID[FK-Person])
- Event(event_name, event_type, event_desc, date, time, audience, has_food, room_number[FK-Location])
- Location(floor, room_number)
- New_items(new_id, new_title, Person_ID[FK-Person], desc, cat_name, condition)
- Organizer(organizer_name, organizer_email)

Many-many Relationships

The many-many relationships were each converted to a relation. The primary keys of each entity and any attributes of the relationship were added to each table. The proposed relations are as follows:

- Register(event_name[FK-Events], Person_ID[FK-Person])
- Volunteer(Person_ID[FK-Person], vol_name[FK-Voluntter_options])
- Fines_on (item_id[FK-Item] , Person_ID[FK-Person], fine_date, fine_amount, fine_payed)
- Items_of_author(author_id [FK-Author] , item_id[FK-Item])
- Items_of_Publisher(publisher_id [FK-Publisher] , item_id[FK-Item])
- Items_of_Producer(producer_id [FK-Producer] , item_id[FK-Item])
- Items_of_Artist(artist_id [FK-Artist], item_id[FK-Item])
- Event_by_Organizer(organizer_email[FK-organizer], event_name[FK-Events])

Proving all relations are in BCNF

In order to avoid redundancy, update and delete anomalies, many of the original relations were decomposed into a collection of relations all in BCNF. The only way the BCNF condition is violated is when there is a bad FD, i.e, a non-trivial FD which is not a superkey.

Below are all possible cases that are present in our relations. By showing that there is no BCNF violation (bad FD) in each cases, we can prove that relations are in BCNF:

Case 1: There are no non-trivial FDs in relation. In this case there can be no bad FDs and consequently no BCNF violations

Case 2: Attribute A is the key, functionally determining all other attributes in relation and all non-trivial FDs contain A. In this case, there can be no BCNF violation since all non-trivial FDs contain A and so is a superkey.

Case 3: Multiple attributes, say A,B ,C together determine all other attributes. In this case A, B,C together are keys and any non-trivial FD is going to contain all of them on the left side and so is in BCNF.

Now we will look at each table:

- Item(item_id, cat_name, title, pub_date,availability, shelf_num, Person_ID$^{FK\text{-}Person}$,borrow_date, due_date,return_date)

  In this relation, item_id is the primary key and functionally determines all other attributes. None of the others are keys as many items(books, audiovisuals etc) can belong to same category, have same name, same publication date, be available at same time, be kept on the same shelf in library(shelf_num), be borrowed by same person(Person_ID$^{FK\text{-}Person}$) and have same borrow_date, due_date,return_date. Hence this falls into **case 2** described above and all non-trivial FDs are going to contain item_id since it is the only key. So there can be no bad FDs and this relation is in BCNF.

- Person(Person_ID, first_name, last_name, address1, address2, city, state, zipCode, phone, email,gender,member_type, mem_startdate, mem_enddate,allow_borrow)
- Personnel(Person_ID$^{FK\text{-}Person}$, job_startdate, job_enddate)

  Similarly, in these relations, person_id is the only key and functionally determines all other attributes. Multiple people can have the same first and last name, and could even live in the same address (eg: grandmother and granddaughter). In this case, it is likely they are the same gender too. Similarly, multiple people could have the same phone and email (again, people from the same family could give the same contact info). Two people can also have the same start and end job or membership dates. Therefore, in order to make the system robust, we have added an artificial attribute- person_ID, that can identify each person uniquely. This again falls into **case 2** described above and all non-trivial FDs are going to contain person_id since it is the only key. So there can be no bad FDs and these relations are in BCNF.

- Books(item_id[FK-Item], edition,online)
- Author(author_id, first_name, last_name)
- Items_of_author(author_id [FK-Author] , item_id[FK-Item])

Item_id determines all the other attributes of Books and is the key as Multiple books can have the same edition and be online. Author_id is the key of Author since multiple authors can have the same name. So these are both in **case 2** described above and all non-trivial FDs are going to contain the only key. So there can be no bad FDs and these relations are in BCNF. In Items_of_author, both attributes are keys and as per the proof of **case 1**, there are no non-trivial FDs and so this is in BCNF. Since authors can write multiple books, the original relation was decomposed into 3 relations- Books, Author and Items_of_author to be in BCNF and remove anomalies.

- Audiovisuals(item_id[FK-Item] , medium)
- Artist(artist_id, Artist_name)
- Producer(producer_id, Producer_name)
- Items_of_Artist(artist_id [FK-Artist], item_id[FK-Item])
- Items_of_Producer(producer_id [FK-Producer] , item_id[FK-Item])

Item_id determines all the other attributes of Audiovisuals and is the key as multiple audiovisuals can be of the same medium (CD/record etc). artist_id is the key of Artist since multiple artists can have the same name. Producer_id is the key of the producer since multiple producers can have the same name. So these are all in **case 2** described above and all non-trivial FDs are going to contain the only key. So there can be no bad FDs and these relations are in BCNF. In Items_of_Artist and Items_of_Producer, both attributes are keys and as per the proof of **case 1**, there can be no bad FDs and these are in BCNF.Since artists and producers can be part of multiple audiovisuals, the original relation was decomposed into 4 relations - Audiovisuals, Artist, producers, Items_of_Producer and Items_of_Artist to be in BCNF and remove anomalies.

- Volunteer_options(vol_name, vol_type, activity_desc ,hours_week)
- Volunteer(Person_ID[FK-Person], vol_name[FK-Voluntter_options])

In Volunteer_options, vol_name determines all other attributes. This is because there can be many volunteering activities of the same type(eg: adult tutoring can be for different ethinic groups but the type will still be adult tutoring) etc. activity_desc and hours_week are also not unique. This again falls into **case 2** described above and all non-trivial FDs are going to contain vol_name since it is the only key. In Volunteer, both attributes are keys and as per the proof of **case 1**, there are no non-trivial FDs and so this is in BCNF. Since multiple people can volunteer for the same activity, the original relation was decomposed into 2 relations- Volunteer and Volunteer_options to be in BCNF and remove anomalies.

- Periodicals(item_id[FK-Item] , issue, volume,online)
- Publisher(publisher_id, Publisher_name)
- Items_of_Publisher(publisher_id [FK-Publisher] , item_id[FK-Item])

Item_id determines all the other attributes of Periodicals and is the key as multiple periodicals can have the same issue number, volume number and can be online. publisher_id is the key of Publisher since it is possible there are multiple publishers with the same name(since publishers can also be individuals not just companies). So these are both in **case 2** described above and all non-trivial FDs are going to contain the only key. So there can be no bad FDs and these relations are in BCNF. In Items_of_Publisher, both attributes make up the key and as per the proof of **case 1**, there are no non-trivial FDs and so this is in BCNF. Since publishers can publish multiple periodicals, the original relation was decomposed into 3 relations- Periodicals, Publisher and Items_of_Publisher to be in BCNF and remove anomalies.

- Location(floor, room_number)
- Event(event_name, event_type, event_desc, date, time, audience, has_food, room_number[FK-Location])
- Event_by_Organizer(organizer_email[FK-organizer], event_name[FK-Events])
- Register(event_name[FK-Events], Person_ID[FK-Person])

In location, room_number is key and determines other attributes(floor). In Event, event_name is key and determines all other attributes. These again fall into **case 2** described above and all non-trivial FDs are going to contain the key of the relation since it is the only key. So there can be no bad FDs and these relations are in BCNF. In Event_by_Organizer, both are needed to make up the key. This is because multiple organizers can co-organize an event and each organizer can host many events. Similarly in Register, each person can register for multiple events and multiple people can register for the same event. These fall into **case 1** scenario as there are no non-trivial FDs and so relation is in BCNF. Since the same location can be used for many events, many events organized by many organizers etc the original relation was decomposed into the above 4 relations-Location,Event_by_Organizer Even and Register to be in BCNF and avoid anomalies.

- Help(help_id, help_type, help_desc, is_urgent, Person_ID[FK-Person])
- New_items(new_id, new_title, Person_ID[FK-Person], desc, cat_name, condition)

In Help, help_id determines all other attributes. This is because there can be multiple requests for the same type of help(eg: researching, writing etc) by the same person. In New_items, new_id is the key. This is because users can donate multiple items with the same name and belonging to the same category(eg: multiple copies of the same book). These again fall into **case 2** described above and all non-trivial FDs are going to contain the key of the relation since it is the only key. So there can be no bad FDs and these relations are in BCNF.

- Fines_on (_ item_id[FK-Item] , Person_ID[FK-Person], fine_date, fine_amount, fine_payed)

In Fines_on, item_id, person_id and fine_date together make up the key and functionally determine the other attributes. This is because the same person could have multiple fines on the same item at different dates. All non-trivial FDs must contain these 3 attributes and so this again falls into **case 3** described above and all non-trivial FDs are going to contain the key of the relation since it is the only key. So there can be no bad FDs and these relations are in BCNF.